Assignment 2

CMPUT 328

Fall 2024

[**Total Weightage:** 8%]

[**Total Points:** 100]

1 Classification with CNNs for CIFAR-10

[70 **points**]

You are going to implement a Convolutional Neural Network (CNN) (`class Net`) to classify the CIFAR-10 dataset. The network architecture is not fixed. You will design the network architecture yourself. You can check out this tutorial to get started.

However, there are some requirements that your network architecture must satisfy:

1. Your network architecture must have at least 4 layers (which is either a convolution or fully connected layer passed into an activation function like Relu, Tanh, etc). Max pooling or strided convolution layers used to downsample activation maps do not count. The number of layers in your network will be very close to the number of activation functions.
2. Must have at least 1 convolution layer.
3. Must have at least 1 max pooling.
4. Must have at least 1 fully connected layer at the end

If your network architecture doesn't satisfy any of the above requirements, **marks will be deducted**. For every requirement that is not satisfied, you lose **10** points. Similar to Assignment 1, observe the metrics on the validation set to check for overfitting.

2 Classification using a pretrained CNN for CIFAR- 10

[30 **points**]

Implement a CNN model using a pretrained architecture (`class PretrainedNet`). Examples include AlexNet, ResNet, etc. This Pytorch documentation might help you out. Save the weights of the final fine-tuned model as **best_model.pth** (`CNN_main.py` does this for you). You will need to submit this!

3 Additional Information

3.1 Template Code

You are given `CNN_main.py` and `CNN_submission.py`. Please do not make any changes to `CNN_main.py` as you will not be submitting this file. `CNN_submission.py` has some template code provided. You are free to make changes to the functions/classes and add your own, but make sure they return what `CNN_main.py` is expecting.

You are free to define your own training/validation splits and transforms in the function `load_dataset(pretrain)`. Some default template code has been provided, that you can modify if you wish to do so. If you need to define your own transformations for the test test, you can pass it to the main file by editing the variable `test_transform`. You can use the variable `pretrain` if you want to define something different for part 2.

3.2 Running the Code

If you have `paramparse` installed, you can use command line arguments.

1. **Part 1:** `python3 CNN_main.py`
2. **Part 2:** `python3 CNN_main.py pretrained=1`
   or
   `python3 CNN_main.py pretrained=1 load_model=1`
   if you want to load the saved checkpoint instead. The runtime displayed when `load_model=1` is the time taken for test set inference. (This is how we'll run your submission for this part while grading!)

3.3 Grading

Please keep in mind that there are no partial marks in this assignment. Your code will not be debugged while grading. Any submission that **fails to run** or does not fall above the **Accuracy** lower-bound will get **no marks**. Exceeding **Runtimes** will result in **penalties**. All runtimes are with respect to Colab GPU.
Runtime penalty: If you exceed the runtime threshold by 10*k%, you will be penalized k%. For example, if you exceeded the runtime by 20%, the incurred penalty will be 2%.

3.3.1 Part 1

1. **Accuracy:** should be minimum 65% to get the maximum score. Score will scale linearly from 55-65% on the test set. Any submission with a test accuracy $< 55\%$ will get no marks.
2. **Runtime:** Should be less than 300 seconds.

3.3.2 Part 2

1. **Accuracy:** should be minimum 90% to get the maximum score. Score will scale linearly from 80-90% on the test set. Any submission with a test accuracy $< 80\%$ will get no marks.
2. **Runtime (inference on test set only):** Should be less than 150 seconds

3.4 Submission Guidelines

Zip `CNN_submission.py` and `best_model.pth` into Assignment2.zip. If you unzip the file, your folder structure should look like:

```
Assignment2
├── CNN_submission.py
└── best_model.pth
```

Do not submit `CNN_main.py` .

3.5 Collaboration Policy

This must be your own work. Do not share or look at the code of other students (whether they are inside or outside the class). You can talk to others in the class about solution ideas (but detailed enough that you are verbally sharing, hearing or seeing the code). You must cite **online resources that were referred to and to** whom you talked with, in the comments of your programs.

Chatgpt is allowed. However, we reserve the right to evaluate any student's submission further through a viva, if we have reason to believe that they do not understand the solution that they have submitted.